10592924

Document made available under the Patent Cooperation Treaty (PCT)

International application number: PCT/US05/009701

International filing date: 23 March 2005 (23.03.2005)

Document type: Certified copy of priority document

Document details: Country/Office: US
Number: 60/555,596
Filing date: 23 March 2004 (23.03.2004)

Date of receipt at the International Bureau: 25 April 2005 (25.04.2005)

Remark: Priority document submitted or transmitted to the International Bureau in compliance with Rule 17.1(a) or (b)

World Intellectual Property Organization (WIPO) - Geneva, Switzerland
Organisation Mondiale de la Propriété Intellectuelle (OMPI) - Genève, Suisse

1309396

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

April 18, 2005

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THE UNITED STATES PATENT AND TRADEMARK OFFICE OF THOSE PAPERS OF THE BELOW IDENTIFIED PATENT APPLICATION THAT MET THE REQUIREMENTS TO BE GRANTED A FILING DATE.

APPLICATION NUMBER: *60/555,596*
FILING DATE: *March 23, 2004*
RELATED PCT APPLICATION NUMBER: *PCT/US05/09701*

UNITED STATES PATENT AND TRADEMARK OFFICE
DEPARTMENT OF COMMERCE

Certified by

Jon W. Dudas

Under Secretary of Commerce
for Intellectual Property
and Director of the United States
Patent and Trademark Office

Please type a plus sign (+) inside this box → +

PTO/SB/16 (02-01)
Approved for use through10/31/2002. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PROVISIONAL APPLICATION FOR PATENT COVER SHEET

This is a request for filing a PROVISIONAL APPLICATION FOR PATENT under 37 CFR 1.53(c).

Express Mail Label No. EV 032702795US

INVENTOR(S)

Given Name (first and middle [if any])	Family Name or Surname	Residence (City and either State or Foreign Country)
Shoubhik	Mukhopadhyay	La Jolla, CA
Debashis	Panigrahi	San Diego, CA

□ *Additional inventors are being named on the ___ separately numbered sheets attached hereto*

TITLE OF THE INVENTION (280 characters max)

SENSOR DATA CORRELATION AND CORRECTION

CORRESPONDENCE ADDRESS

Direct all correspondence to:

☒ Customer Number 24978 → *Place Customer Number Bar Code Label here*

OR *Type Customer Number here*

□ Firm *or* Individual Name					
Address					
Address					
City		State		ZIP	
Country		Telephone		Fax	

ENCLOSED APPLICATION PARTS *(check all that apply)*

☒ Specification *Number of Pages* 15 □ CD(s), Number

□ Drawing(s) *Number of Sheets* □ Other (specify)

□ Application Data Sheet. See 37 CFR 1.76

METHOD OF PAYMENT OF FILING FEES FOR THIS PROVISIONAL APPLICATION FOR PATENT

☒ Applicant claims small entity status. See 37 CFR 1.27.

☒ A check or money order is enclosed to cover the filing fees

☒ The Commissioner is hereby authorized to charge filing fees or credit any overpayment to Deposit Account Number: 07-2069

□ Payment by credit card. Form PTO-2038 is attached.

FILING FEE AMOUNT ($) $80.00

The invention was made by an agency of the United States Government or under a contract with an agency of the United States Government.

☒ No.

□ Yes, the name of the U.S. Government agency and the Government contract number are: ______

Respectfully submitted,

SIGNATURE

Date 3 23 04

TYPED or PRINTED NAME Steven P. Fallon

REGISTRATION NO. *(if appropriate)* 35,132

Docket Number: 0321.70116

TELEPHONE (312) 360-0080

USE ONLY FOR FILING A PROVISIONAL APPI

This collection of information is required by 37 CFR 1.51. The information is used by th
provisional application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14.
complete, including gathering, preparing, and submitting the complete provisional applica
the individual case. Any comments on the amount of time you require to complete this fo
should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S
20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SE
Commissioner for Patents, Washington, D.C. 20231.

I hereby certify that this paper is being deposited with the United States Postal Service as Express Mail in an envelope addressed to: M.S. PROVISIONAL PATENT APPLICATION, COMMISSIONER FOR PATENTS, P.O. Box 1450, Alexandria, VA 22313-1450, on this date.

March 23, 2004
Date

Daniel Car

Express Mail No. EV032702795US

PROVISIONAL APPLICATION COVER SHEET

Additional Page

PTO/SB/16 (02-01)
Approved for use through 10/31/2002. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

Docket Number	0321.70116	Type a plus sign (+) inside this box →	+

INVENTOR(S)/APPLICANT(S)		
Given Name (first and middle [if any])	Family or Surname	Residence (City and either State or Foreign Country)
Sujit	Deyi	La Jolla, CA

Number 2 of 2

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

I hereby certify that this paper is being deposited with the United States Postal Service as Express Mail in an envelope addressed to: M.S. PROVISIONAL PATENT APPLICATION, COMMISSIONER FOR PATENTS, P.O. Box 1450, Alexandria, VA 22313-1450, on this date.

March 23, 2004
Date
Express Mail No. EV032702795US

SENSOR DATA CORRELATION AND CORRECTION

FIELD OF THE INVENTION

The field of the invention is sensor networks.

BACKGROUND

Many systems, for example in manufacturing, testing, monitoring, collect data from a number of sensors. Wireless data collection is an important technique due to the flexibility provided by wireless networks. Even more so than in other applications that use wireless data transfer, providing reliable data collection is a paramount concern in sensor networks, as the data is collected, processed, and used to make decisions in a machine-to-machine data collection framework. There are well known problems with wireless data transfer relating to the reliability and correction of data. Most techniques for gauging reliability place a high overhead on the collection. Typical existing reliability methods may add redundant hardware or transmit extra data at the source to correct for data corrupted in the circuits or the communication channels respectively. This makes the typical methods prohibitively expensive to be used with heavily constrained sensor nodes. To address failures in circuits and communication channels, they incur high overheads in terms of energy budget, and design and manufacturing cost in the sensor nodes.

Example prior methods include methods to correct soft errors in hardware as well as those to correct bit detection errors on a wireless communication channel. Techniques for the former include both circuit and module level approaches, e.g. triple modular redundancy and error correction coding in hardware. Techniques for the communication channel include parity-based forward error correction (FEC) coding techniques like channel coding, and retransmission-based techniques like ARQ.

DESCRIPTION OF PREFERRED EMBODIMENTS

The invention provides reliability with minimal cost of error protection, i.e. the cost of sensor nodes and communication overhead. In embodiments of the invention errors run-time error correction is conducted, either within the circuits of the sensor nodes or over the communication channel, with no design or operational overhead on the sensor node.

According to the invention, information contained in the sensor data itself, for example, is used to achieve data checking and correction. Embodiments of the invention use information about the temporal correlations in sensor data, the goals of the application, and its vulnerability to various errors. Sensor data generally exhibits redundancy over a temporal period. The inherent redundancy of the sensor data may be leveraged to make possible a high degree of reliability in data collection, without imposing overheads on sensor nodes at the expense of nominal buffer requirements at the data aggregator nodes, which are much less cost/energy constrained.

A network embodiment of the invention includes a plurality of sensor nodes that wirelessly communicate data to an aggregator node. The inherent redundancy sensor data is utilized to perform error correction at the site of data processing, which can, for example, be a data aggregator node with more computational, storage and energy resources than the sensor nodes.

Embodiments of the invention also include an aggregator node for use in a wireless network. The aggregator node as a predictive model based upon off-line analysis of sensor data from nodes of a network. When on-line, the aggregator node conducts a reliability check at run-time using the predictive model to check inherent redundancies in the sensor data for reliability, and to make error correction decisions. Methods of the invention include collecting data off-line for an inherent sensor data redundancy predictive model, and applying the model on-line at run time, preferably at a collection point such as an aggregator node for sensor data received over the wireless network.

Example embodiments will now be discussed. Additional embodiments are described in an attachment, and incorporated herein.

An embodiment of the invention is a method, for example implemented in software, for correction of transient errors in sensor data at an aggregation node where aggregation and filtering of the sensor data occurs in a sensor data network. The method achieves run-time correction of data received from sensor nodes over wireless communication channels, without imposing any design or material cost or performance overheads on the sensor nodes. The overhead incurred is solely in terms of storage and computation costs at gateway node(s) that buffer data for the aggregation node, and can be tuned to the performance requirements of the application and resource constraints at the aggregator. The method uses steps to identify and use redundancies within the sensor data to correct against transient errors. An offline analysis of redundancy within sensor data captures correlation properties in a predictive model. The correlation models are used for predictive correction of the data online. The FIG. 1 method, illustrated below, is consistent with this embodiment.

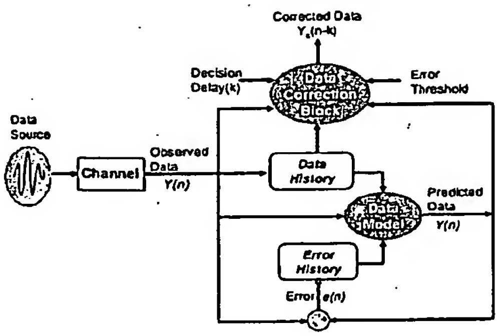


FIG. 1

With reference to FIG. 1, a predictive model is constructed by pre-processing of initially collected data. A model chosen should be rich enough for the predictions to match the data generation process. The model should allow a process of prediction that is efficient in terms of resource consumption and complexity to meet any performance requirements of the gateway. The modeling itself may be more computationally complex since it will not be performed online. The choice of model given the above requirements will depend mostly on the level and nature of temporal correlation in the data. An example model, used in experiments to test the invention, is the Auto-regressive moving average (ARMA) model. This is a linear predictive model which uses the history of previous observations as well as that of prediction performance, as shown in FIG. 1.

As shown in Figure 1, the predictive model explained above is used at run-time for computing the likely value of the next reading, and the data correction method determines whether the value obtained from the sensor or that provided by the model will be used for future use. Steps for this include maintaining a history of observations and using the model to predict the future value from the history. Once the next observed value is received from the sensor node, it is decided which of these candidate values to record.

An important issue in performing the prediction-based correction choosing to handle mismatches between a predicted and observed value at the receiver, which may have been caused by a genuine error or by departure of the source's behavior from the model. In embodiments of the invention this decision is made based on the past samples as well as a few samples observed afterwards. This is illustrated in Figure 1 in the delay parameter (*K*) used. As shown in Figure 1, the observed values up to *Y(n)* and the corresponding predictions up to *Y'(n)* are used, after a delay of *K* samples, to report the corrected value *Yc(n-K)*. For every sample of the sensor value observed, the correction step compares it with the value predicted from the model and past history, and attempts to report the value closer to the actual expected observation. The delaying of this decision allows the step to consider the effect of any choice it makes on the prediction accuracy for the *K* samples following it. This is implemented in the step using the prediction history tree (PHT), which contains the possible predicted values and the corresponding prediction errors for the past *K* samples. The PHT has a depth of K+1, and represents the various potential values for the last K samples, i.e. *Yc(i)* where *i=n-K:n-1*.

Figure 2 shows an example of a PHT for K=3. Each node in any level j of the tree represents a possible value of *Yc(n-K+ j -1)*, with the root node (level 0) denoting the value already chosen for *Yc(n-K-1)*. Every node has two outgoing paths, labeled 0 and 1. These represent the choices of Y (observed value) and Y' (predicted value) respectively for the sample following it. Thus, every path from root to a leaf in level K+1 denotes a series of up to 2K choices leading to a sequence of values *Yc(n-K:n-1)*. The nodes of the PHT in the figure have been annotated with the possible values contained in them, where *Y'(n-k|010)* mean the predicted value obtained after following the path corresponding to the choices of 010 from the root node.

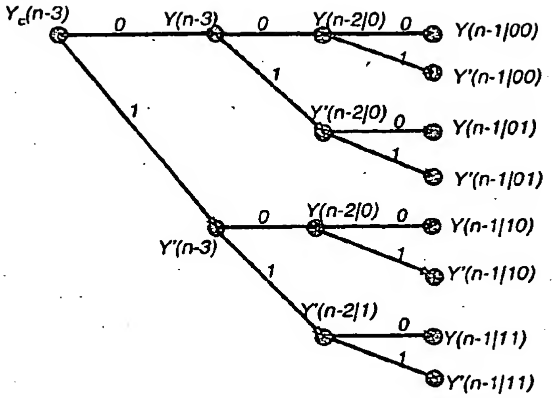


FIG. 2

The pseudo-code of the step used to correct errors at the receiver using the PHT is shown in Figure 3. Before receiving *Y(n)* at time n, up to 2K possible values of *Y'(n)* are computed, one for each path from the root to every leaf node. After *Y(n)* is received, the value of *Yc(n-K)* is estimated, the prediction errors for the different paths (E) are compared, and based on the minimum accumulated path error (*PathErr*), one of the branches from the root are chosen to generate the PHT for the next level. The sub-tree rooted at the other branch from the root is discarded, and the remaining tree is extended another level by adding one or two children (observed *Y(n)* and the prediction *Y'(n)* for that path) to each leaf. For efficiency, the error threshold value is used to avoid adding new *Y'(n)* values if *E(n)* is below ETH. This means that if that particular node becomes the root after N steps, the observed value Y should be used for Yc. So the tree structure will often not be fully populated.

```
begin
for each n,
        for each path i from root to leaf in PHT
        define Ytemp(1:n-1) =[Yc(1:n-K-1)|Ypht(n-K:n-1)]
        Y'(n, i)  = predict(Ytemp(1:n-1), E(1:n-1))
        E(n, i)  = Y(n) - Y'(n, i)
        PathErr(i) =
$\sum_{j=n-K}^{n} E^2(j,i)$
        end
        find $i = I_{min}$ which minimizes $PathErr(i)$;
        $<Yc(n\text{-}K,)E(n\text{-}K)>$ = updatePHT($I_{min},$ $Y'(n, i)$, $\dot{Y}(n)$)
end
end

updatePHT($i$, $Y'(n', j)$, $Y(n)$):
begin
        find $s$ = level 1 node containing path i
        $<y, e>$ = Y and E values of node $s$
        PHT ← subtree of PHT rooted in $s$
        to each leaf node $j$ of new PHT, add 1st child $Y(n)$,
        and if $(|E(n,j)| > ETH)$ add 2nd child $Y'(n, j)$
        return $<y,e>$
end
```

FIG. 3 Correction step Pseudocode

The choice of *K* determines, apart from the delay in reporting the corrected values, the level of correction achieved by the algorithm under particular given data and error characteristics. The storage and computational complexity of the method also depend directly on the parameter K, since it determines the amount of history information used for correcting each sample. Since a goal of the method is to distinguish between modeling errors and real random errors occurring in the sensor node and/or the channel, the optimum choice of K depends on the properties of the errors as well as the performance of the modeling technique used. Potentially, it is also possible to trade off correction accuracy against performance and resources by varying K, and match them to the application requirements and constraints of the micro-gateways.

Different depths of prediction histories may be used depending on the application's delay sensitivity, the relative error levels and the resource constraints on the receiving node

While specific embodiments of the present invention have been shown and described, it should be understood that other modifications, substitutions and alternatives are apparent to one of ordinary skill in the art. An example claim is presented. Further embodiments are illustrated in an attachment.

Exemplary Claims:

1. A method for collecting sensor data over a wireless network, the method comprising steps of:

in an offline mode, by a device other than a sensor node,

wirelessly collecting initial sensor data from one or more sensor nodes in the wireless network;

pre-processing of initial sensor data to determine a level of inherent temporal redundancy in the data;

developing a predictive model based upon inherent temporal redundancy in the initial sensor data;

in an online mode, by a device other than a sensor node,

computing the likely value of a next sensor reading from a sensor node in the wireless network based upon the predictive model;

determining whether the value received from the sensor node is reliable with respect to the likely value, and, if not correcting the value received from the sensor node.

Data aware, Low cost Error correction for Wireless Sensor Networks[†]

Shoubhik Mukhopadhyay, Debashis Panigrahi, Sujit Dey
Mobile Embedded Systems Design and Test Lab
Department of Electrical and Computer Engineering
University of California, San Diego
{shoubhik, dpani, dey}@ece.ucsd.edu

Abstract—**One of the main challenges in adoption and deployment of wireless networked sensing applications is ensuring reliable sensor data collection and aggregation, while satisfying the low-cost, low-energy operating constraints of such applications. A wireless sensor network is inherently vulnerable to different sources of unreliability resulting in transient failures. Existing reliability techniques that address transient failures in circuits and communication channels incur prohibitively high energy, bandwidth and cost overheads in the sensor nodes. In this paper we investigate application-level error correction techniques for sensor networks that exploit the properties of sensor data to eliminate any overhead on the sensor nodes, at the expense of nominal buffer requirements at the data aggregator nodes, which are much less cost/energy constrained. Our approach involves use of spatio-temporal correlations in sensor data, the goals of the application, and its vulnerability to various errors. We present our error-correction algorithm and evaluate it through simulations using real and synthetic sensor data. Experimental results validate the feasibility of our approach to provide high degree of reliability in sensor data aggregation, without imposing overheads on sensor nodes.**

Keywords-wireless sensor networks; reliability; application-specific error correction

I. INTRODUCTION

The convergence of techniques for sensing, communication, and processing has led to the emergence of wireless sensor networks. The availability of these sensor networks enables sensing and monitoring of the physical world, with possible applications ranging from environment and traffic monitoring to industrial engineering process control. Recently, large scale sensing has become feasible with the use of low-cost, low-energy wireless sensor nodes. However, such devices are vulnerable to various sources of errors as discussed below. Hence, providing reliable data collection and aggregation has become paramount concern for deploying such sensor applications.

A wireless network of sensor nodes is inherently exposed to various sources of unreliability, such as unreliable communication channels, node failures, malicious tampering of nodes and eavesdropping. The sources of unreliability can be classified into two categories: (1) faults that change behavior permanently, and (2) failures that lead to transient deviations from normal behavior, termed as *soft failures* in this paper.

The permanent faults include failures due to unavailability of energy resource, calibration errors after prolonged use, and loss of wireless coverage. Soft failures occur in wireless channels as transient errors, caused by noise from various sources, such as thermal noise at the receiver, channel interference and multi-path fading effects. Additionally, the use of aggressive design technologies such as deep-sub-micron (DSM) and ultra-deep-sub-micron (UDSM), to reduce the cost of each node further exposes the nodes to different types of transient errors in computations and sensing. The primary sources of such transient errors in nodes are ionizing particle strikes and electromagnetic noise from various sources such as crosstalk and transmission line effects [1].

A few techniques have been recently proposed to address unreliability due to permanent failures in sensor networks. These include providing reliability at the transport-level against communication failures [2, 3], ad-hoc routing techniques that can cope with dynamic node failures [4], and in-network data aggregation and fusion [5]. However, they are designed for ad-hoc sensor networks, where the sensor nodes have to incur the overhead of implementing the reliability mechanisms. A hierarchical network architecture [6] in which aggregator nodes collect data from very constrained sensor nodes is easier to setup and maintain. But in such architectures, the cost and energy overheads imposed by the above schemes on the sensor nodes are prohibitive. Moreover, such techniques are agnostic towards properties of sensor data and goals of applications. We show here that by using information about the sensor data and applications, the reliability can be improved without imposing any overhead on the sensor nodes.

Soft failures have been addressed in other fields of research by methods such as circuit-level enhancement techniques for computation unreliability [1], and channel coding techniques to enable reliability during communication [7]. However, due to the resource limitations such as size, cost and energy resources of sensor nodes, it is prohibitive to deploy the above conventional reliability techniques in sensor networks. For example, as shown later in this paper, Reed-Solomon based coding [8] for communication errors leads to a transmission overhead of 40-160% thus constraining the low-energy nodes

[†]This work is supported by the Center for Wireless Communications (CWC), University of California at San Diego

severely. Similarly, circuit hardening techniques lead to large overheads in area and timing as stated in [1].

A. *Paper Contributions and Overview*

We observe that data collected by sensor nodes shows a significant amount of spatial and temporal correlation. Additionally, the aggregation operations performed by applications on collected data have varying levels of vulnerability to erroneous data. Based on the above observations, we focus on developing low-cost error correction methods for soft failures using the properties of data captured in a data prediction model.

In this paper, we present an algorithm that uses data predictions to filter out errors caused by soft failures. While data predictions filter out the majority of errors in the observed values, it is possible that the predictions may not always track the data process variations correctly. In such cases, to increase the effectiveness of our algorithm, we also provision for delaying the reporting of data within the application's delay constraints. The delayed reporting allows us to use observed values in the next few samples to guide the choice of correct value between the predicted and observed value. The algorithm can be tuned to the computational resources available at the receiver and the application's delay requirements. Additionally, our approach does not require any overhead in transmission by the sensor nodes. Instead, most of the error correction functionalities are pushed to the receiver, which is less resource-constrained than the sensor nodes. We demonstrate that our proposed approach reduces the cost of soft failure protection significantly compared to other existing approaches.

The rest of the paper is organized as follows. In the next section, we motivate the need for using data properties to guard against soft failures. We describe our approach to provide reliability for soft-failures in Section 3. Experimental evaluation of the proposed technique follows next. Finally, we provide summary of our proposed approach and discuss scope for future extensions.

II. Motivation

In this section, we motivate how application level properties can be used to reduce cost of reliability for soft failures such as transient errors in communication channels and sensor nodes.

Let us consider a light sensor monitoring the variation in luminance value inside a room for one day. Let as assume the sensor application is interested to observe the variation of luminance value over a time window of 60 seconds. Figure 1 shows plots of luminance data collected by light sensors deployed in our lab. It can be seen from the plots that there is a strong temporal correlation in the data collected. We plan to capture temporal correlation to predict next value based on past samples, and hence correct the presence of transient errors. Additionally, since the predictions may not track the variations in the data perfectly, we plan to use a data correction scheme that chooses a data value based on the past data samples as well as a few samples observed afterwards.

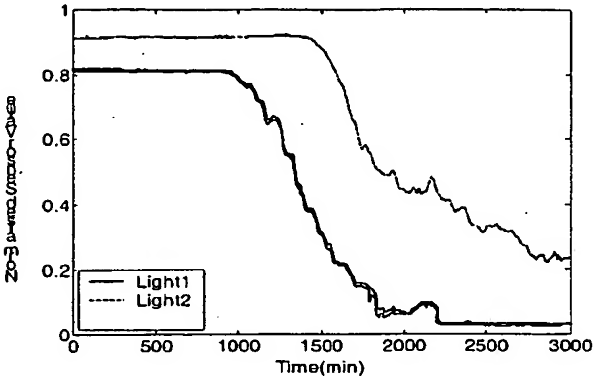


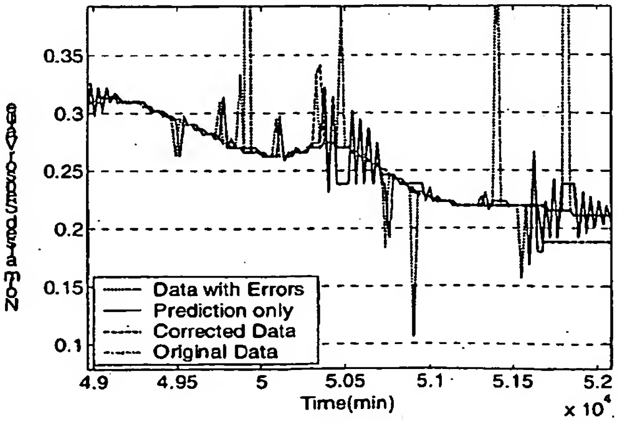
Figure 1 Temporal Correlation in Sensor Data

To compare different approaches, in Figure 2 we plot the original data, the data with communication errors (at a bit-error-rate of 10^{-2}), the data corrected using predictions only, and data corrected by our proposed algorithm. It can be seen from the figure that, the use of prediction decreases the effect of errors, but does not match with the original data satisfactorily. However, data produced by our error correction approach matches perfectly to the original data, with an associated error of only 4%. This shows the promise of the use of data properties to provide reduced errors with no overhead at the sensor nodes. This will be particularly useful within network architectures which consist of large numbers of cheap and light sensor nodes managed by aggregator nodes with comparatively larger energy and resource budgets.

III. Data-aware Error Correction

In this section we present our technique of correction of transient errors in the sensed data. The framework consists of three main processes: (1) a model of the data generation process is constructed off-line based on the correlation observed in samples of sensor data; (2) this model is used during data acquisition for online prediction of data; and (3) an application-aware predictive correction block uses the

Figure 2 Data Prediction and Correction Example



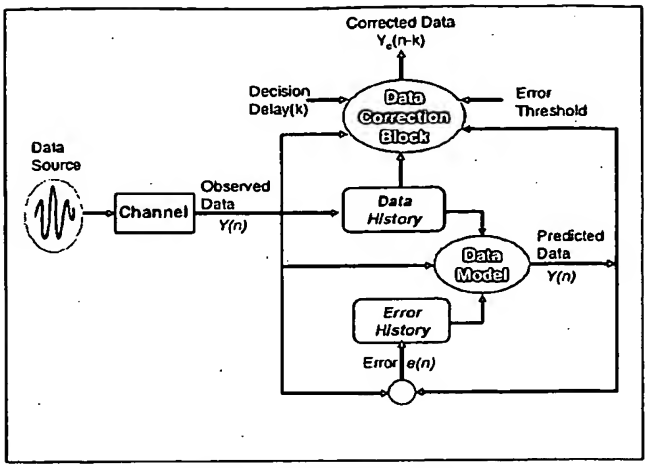


Figure 3 Algorithm Schematic

prediction history to correct errors in the data. Figure 3 illustrates the online operation of this method. The observed data, received over the wireless channel, contains errors originating at the source node as well as those introduced in the channel. A running history is maintained of the observed values as well as prediction errors. These histories are then used together with the data model created in step 1 to generate predictions, and possibly update the model as well. The data correction block (step 3) uses these predictions to estimate the correct data.

The operation of the error correction block is independent of the data model used for prediction. A variety of modeling techniques can thus be used to represent data correlation properties. However, the performance of the correction algorithm depends on the accuracy of modeling and efficiency of the prediction.

A. Data Correction Algorithm

A key issue in performing prediction-based data correction at the receiver is that of distinguishing between different types of errors. A mismatch between a predicted and observed value at the receiver may have been caused by a genuine error or by departure of the source's behavior from the model, and it should be handled differently in these two cases. One way to distinguish between them is the use of prior knowledge of their statistical properties. For example, single-bit errors occurring in MSBs of sensor samples will very likely result a sharp jump in sensor values, while changes in the physical process generating the data will be slower and possible for the model to track successfully. One way to aid help making this decision is delay it and consider the samples following the aberrant one. This is implemented by the prediction history tree in the algorithm. Different depths of prediction histories may be used depending on the application's delay sensitivity, the relative error levels and the resource constraints on the receiving node.

In the framework for the data correction algorithm shown in Figure 3, *Y* represents the sequence of observed values of sensor data, *Y'* represents the results of the prediction block, and *Yc* represents the corrected values from the data correction block (step 3). The data correction block uses the prediction block in the process of correcting errors by generating and storing different possible version of the history of recent predictions as discussed below. At any point in time *n*, given *Y(n)*, the data correction block computes a corrected value *Yc(n-K)*, where *K*, called the decision delay, represents the depth of the prediction history maintained for a posteriori correction. The size of the prediction history can be somewhat reduced by assuming very small variations from the predictions to be due to randomness in the sensed physical process rather than transient errors. This is implemented by a second control parameter, called the error threshold (*ETH*), as discussed below.

The history of observed and predicted values and the corresponding prediction errors is stored in the prediction history tree (PHT). The prediction errors corresponding to each node's value in the PHT is stored in a parallel error history tree, which is always maintained in sync with the PHT by performing same update operations on both. The PHT has a depth of *K+1*, and represents the various potential values for the last K samples, i.e. *Yc(i)* where *i=n-K:n-1*. Figure 4 shows an example of a PHT for *K=3*. Each node in any level *j* of the tree represents a possible value of $Y_c(n\text{-}K+j\text{-}1)$, with the root node (level 0) denoting the value already chosen for *Yc(n-K-1)*. Every node has two outgoing paths, labeled 0 and 1. These represent the choices of *Y* (observed value) and *Y'* (predicted value) respectively for the sample following it. Thus, every path from root to a leaf in level *K+1* denotes a series of up to 2^K choices leading to a sequence of values *Yc(n-K:n-1)*. The nodes of the PHT in the figure have been annotated with the possible values contained in them, where *Y'(n-k|010)* mean the predicted value obtained after following the path corresponding to the choices of *010* from the root node.

The pseudo-code of the algorithm used to correct errors at the receiver using the PHT is shown in Figure 5. Before receiving *Y(n)* at time n, up to 2^K possible values of *Y'(n)* are computed, one for each path from the root to every leaf node. After *Y(n)* is received, the value of *Yc(n-K)* is estimated, the

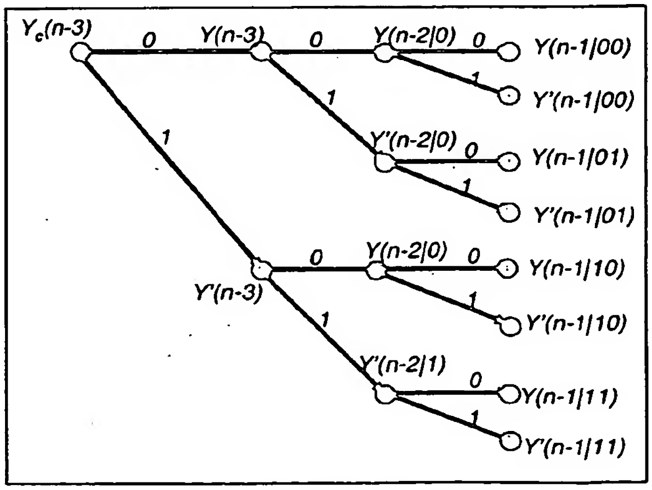


Figure 4 Prediction History Tree for K=3

prediction errors for the different paths (E) are compared, and based on the minimum accumulated path error (PathErr), one of the branches from the root are chosen to generate The subtree rooted at the other branch from the root is discarded, and the remaining tree is extended another level by adding one or two children (observed *Y(n)* and the prediction *Y'(n)* for that path) to each leaf. For efficiency, the error threshold value is used to avoid adding new *Y'(n)* values if *E(n)* is below *ETH*. This means that if that particular node becomes the root after *N* steps, the observed value *Y* should be used for *Yc*. So the tree structure will often not be fully populated.

The accuracy of the data correction in this approach depends on the accuracy of the modeling stage. The performance of the correction method also depends on that of the prediction algorithm, which is invoked for each path for every sample to predict the next value in that sequence. The primary resource consumed by the correction block is storage, the space complexity being $O(2^K)$ for the PHT.

B. Data Modeling:

The applicability of this approach relies upon representative samples of the sensor data being available in order to build accurate predictive models. This includes all applications that observe the current state of some physical process and monitor them for known or unknown variations. Various techniques exist for modeling unknown systems, and since the initial identification is to be performed off-line, the computational complexity of the identification process does not affect the resources of the system.

Our predictive correction approach is not effective for a class of applications where the purpose is to observe and model a physical phenomenon[9], since in such cases it is not possible to pre-compute the data models. However, there are many applications that provide models of various degrees of accuracy and the algorithm can be extended to update the models online based on observations or to update the nodes with new models at run-time. In addition to exploiting temporal correlation of data, another natural extension of this approach will be using the spatial correlation properties between multiple sensor nodes.

```
begin
for each n,
        for each path i from root to leaf in PHT
        define Ytemp(1:n-1) = [Yc(1:n-K-1)|Ypht(n-K:n-1)]
        Y'(n, i) = predict(Ytemp(1:n-1), E(1:n-1))
        E(n, i) = Y(n) - Y'(n, i)
        PathErr(i) = $\sum_{j=n-K}^{n} E^2(j,i)$
        end
        find $i = i_{min}$ which minimizes PathErr(i);
        <Yc(n-K,)E(n-K)> = updatePHT($i_{min}$, Y'(n, i), Y(n))
end
end

updatePHT(i, Y'(n', j), Y(n)):
begin
        find s = level 1 node containing path i
        <y, e> = Y and E values of node s
        PHT ← subtree of PHT rooted in s
        to each leaf node j of new PHT, add 1st child Y(n),
        and if (|E(n,j)| > ETH) add 2nd child Y'(n, j)
        return <y,e>.
end
```

Figure 5 Correction Algorithm Pseudocode

IV. Experimental Evaluation

In this section, we present the evaluation of the performance of the proposed approach. First, we present the simulation setup and data modeling used in the evaluation. Next, we evaluate our approach with different input parameter values, data modeling inaccuracy, and compare with other existing techniques of error correction.

Simulation & Modeling Framework:

Our algorithm was evaluated using a Matlab-based simulation setup with our data correction algorithm as well as the data modeling incorporated. As input data set for our simulations, we use data read from a sensor as well as those synthesized based on specific models. The sensor data are collected from an indoor light sensor sampled at 0.1Hz into 8bit samples. Errors occurring in the communication channel as well as in logic and memory circuits of the nodes are simulated as uniformly distributed random bit errors, in the range of 10^{-4} to 10^{-2}.

To model the data properties, we have limited the exploration of the models by assuming that the data source is stochastically stationary. With this assumption, we used Auto-Regressive Moving Average (ARMA) models [10] to capture the self-similarity of the observations and provide a measure of the dependency of the system's current state on its past. The key step in effective modeling is identification of the system's order, i.e. the number of past values and error history to be used for computing the new predicted value. This was performed by using the minimum final prediction error criterion [11] to choose N over a fixed range (1-10) to fit ARMA(N, N-1) models. Although within our overall scheme, it is possible to keep updating the initially estimated model at run-time, in our implementation we have performed only offline modeling during system identification stage.

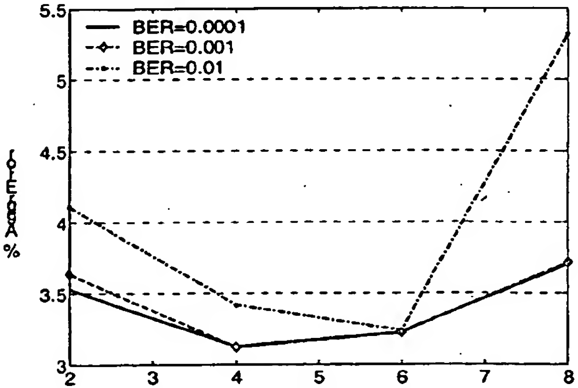


Figure. 6 Aggregation Errors vs. DD at different BERs for Light Sensor Data

Effect of Delay on Accuracy:

We investigate the possible tradeoffs in correction accuracy vs. overheads of delay and memory by varying one input parameter of the Data Prediction Block, namely decision delay (DD) parameter, described in Section 3. As discussed in the previous section, the parameter DD directly affects the delay in the data correction step and the storage requirements. It is difficult to find a quality metric that captures the effect of errors in data across different types of sensing applications. We use the root mean square sum of residual errors of the of the aggregated sensor values in the output after correction. The aggregation function used is moving average function with a 1min window.

The variation of aggregation error against the DD parameter for various values of BER is presented in Figure. 6. From the figure, it can be observed that for lower values of DD, the aggregation error decreases with increase in DD. For instance for a BER of 10^{-3} the error is reduced from 3.7% to 3.2% as DD is increased from 2 to 4. The increase in DD leads to a better data correction with the availability of more neighboring data samples. However, after a certain point, a increase in DD leads to an increase in aggregation error. This is probably because the extra samples used at higher DD are themselves more likely to be erroneous, which leads to a wrong decision by the correction block. E.g. for 8bit samples (ignoring the frame headers), a BER of $2x10^{-2}$ implies an expected error in every 6th sample. In such a case, a DD of 6 or more will reduce accuracy. Hence better accuracy can be obtained by adjusting the DD parameter run time based on an estimate of the current error level.

Effect of modeling Error on Accuracy:

Next, we evaluate the effect of modeling on the accuracy of our approach. To study the dependency of the prediction accuracy on that of modeling, we use a set of data synthesized using an ARMA(9,8) model and present the performance when an ARMA(7,6) model is used to predict and correct it. The effect of modeling accuracy is presented in Figure 7. The figure shows the aggregation error achieved by the two models, i.e. ARMA(7,6) and ARMA(9,8), for different BER conditions and DD values. It can be observed that modeling accuracy affects the accuracy of our algorithm; i.e. the aggregation error for ARMA(9,8) is lower than that if we use a ARMA(7,6) which is an inaccurate model of the data in this case.

It is important to observe that the aggregation error of our approach is very small compared to data modeling inaccuracy as we do not base our decision on one predication, rather a collection of samples. For instance, the modeling inaccuracy of ARMA(7,6) is 4%, however observed degradation in error is 1%.

Comparison with Channel Coding:

We also present a comparison of the performance of our approach with channel coding, one of the schemes to protect against transient communication errors. The graph in Figure 8 shows the aggregation errors in the light sensor data obtained using our approach (DAPC), and Reed-Solomon (RS) coding with different coding rates. It can be seen that effect of different levels of channel coding on aggregation error depends on the BER conditions. While RS coding can lead to very accurate results by reducing the code rate, the communication overhead is very high. For instance, for a BER of 0.01, our scheme achieves an aggregation error of 3.1%, whereas to achieve that level of error by RS coding we need to use a coding rate of 0.53 that leads to an overhead of 87% in transmitted data. Moreover, to protect against node errors, channel coding needs to be supplemented by circuit-level techniques with large area and energy overheads [1].

TABLE I. presents a comparison of transmission and storage costs incurred by RS coding and DAPC for different levels of aggregation error tolerances. To represent overhead, we use the transmission energy for RS coding, and transmission energy and storage requirements for our scheme (DAPC). For each target aggregation error, we select the coding rate of RS coding and the DD parameter of DAPC that leads to the least overhead while achieving the target error rate. The transmission of energy is computed using the energy

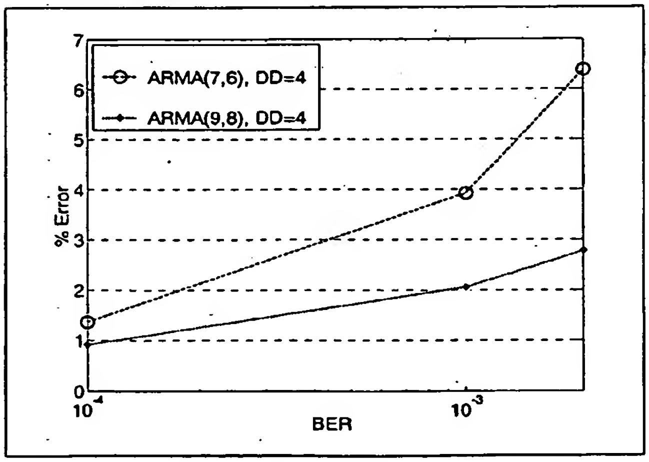


Figure 7 Effect of Modeling Accuracy

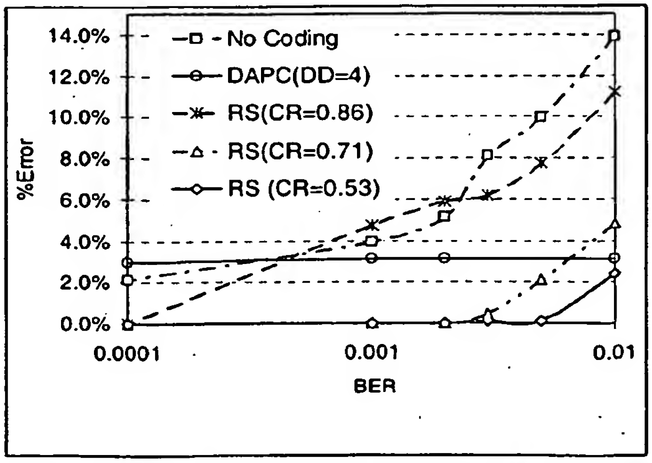


Figure 8 Aggregation Errors vs. BER in Light Data for Different Correction Methods

TABLE I. AGGREGATION ERRORS AND OVERHEADS

Bit Error Rate	Target Error Rate (%)	Original Tx Energy (joules)	Tx Energy/ % overhead using RS (joules)	Table Size in DAPC (bytes)
0.001	5%	0.16	0.19 / 17%	16
0.001	3%	0.16	0.22 / 40%	128
0.01	5%	0.16	0.22 /40%	16
0.01	3%	0.16	0.30 / 87%	128

model for a MICA mote transmitting 10000 8-bit samples. We also present the storage required for the PHT for the corresponding DD value. It is observed that our scheme achieves the target error rate using much less transmission energy at the sensor node incurring minimal storage costs at the receiver. For instance, to achieve a target error rate of 3% at BER=10^{-2}, our scheme achieves a saving of 87%.

V. Conclusion and Future Work

In this paper, we presented application-level data-aware error correction, which exploits the correlation of sensor data to perform partial error correction at very low cost. We are currently investigating sensor data used by different applications for a better understanding of the data models and considering the possibility of performing the data modeling online. We are also extending the framework presented here to exploit spatial correlation for making multi-sensor data aggregation functions reliable. Another possible improvement is to extend this idea to improve the robustness of aggregation against permanent failures.

References

[1] Y. Zhao and S. Dey, "Separate Dual Transistor Registor-an Circuit Solution for on-line Testing of Transient Errors in UDSM-IC," *in Proceedings of* Intl. On-line Testing Symposium 2003, Kos Island, Greece, 2003.

[2] C. Y. Wan, et al., "PSFQ: a reliable transport protocol for wireless sensor networks," *in Proceedings of* 1st ACM international workshop on Wireless sensor networks and applications, Atlanta, Georgia, USA, 2002.

[3] F. Stann and J. Heidemann, "RMST: Reliable Data Transport in Sensor Networks," *in Proceedings of* First International Workshop on Sensor Net Protocols and Applications (SNPA '03), Alaska, 2003.

[4] C. Intanagonwiwat, et al., "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *in Proceedings of* Sixth Annual International Conference on Mobile Computing and Networking (MobiCOM), Boston, MA, USA., 2000.

[5] E. Elnahrawy and B. Nath, "Cleaning and Querying Noisy Sensors," *in Proceedings of* Second ACM Intl. Workshop on Wireless Sensor Networks and Applications (WSNA), San Diego, CA, 2003.

[6] R. Kumar, et al., "Computation Hierarchy for In-network Processing," *in Proceedings of* 2nd Intl Workshop on Wireless Sensor Networks and Applications (WSNA), San Diego, 2003.

[7] P. Lettieri, et al., "Low power error control for wireless links," *in Proceedings of* 3rd annual ACM/IEEE Intl. conference on Mobile computing and networking (MOBICOM), 1997.

[8] S. B. Wicker, Error Control Systems for Digital Commn and Storage: Prentice Hall, 1995.

[9] A. Mainwaring, et al., "Wireless Sensor Networks for Habitat Monitoring," *in Proceedings of* First ACM Workshop on Wireless Sensor Networks and Applications(WSNA), Atlanta, GA, USA, 2002.

[10] S. M. Pandit and S. Wu, *Time Series and System Analysis*: John Wiley and Sons, 1983.

[11] H. Akaike, "Fitting autoregressive models for prediction," *Ann. Inst. Stat. Math*, vol. 21, pp. 243-247, 1969.